BE230E1
Software Programming Guide

Rev. 1.0

Revision History

Version	Note	Date
0.1	Initial version	2016/10/25
0.5	Add Bolymin API definitions	2016/11/15
1.0	1st DVT release	2016/12/05

Table of Content

1 Setup Software Development Environment

1.1 Linux Host Machine for Software Development

To do Linux development with the SDK for BE230E1 embedded product, you'll need a host PC running Linux OS. The Linux host is generally much faster and has a lot more memory (both RAM and hard disk space) than the typical embedded system so that it's usually used to install specific cross compiler to build executable programs for target device. There're many choices of Linux distributions for your Linux PC and we choose 64-bit Ubuntu 14.04 LTS (or later version), running either natively or in a Virtual Machine. Note that **it's necessary to use 64-bit Linux OS** to develop application programs on BE230E1 device.

1.2 Install SDK on Linux Host

Prerequisite

Before using toolchain, please check if some dependent packages are installed in your Linux Host, or install them by

```
$sudo apt-get install xinetd tftpd nfs-kernel-server minicom
build-essential libncurses5-dev autoconf automake
```

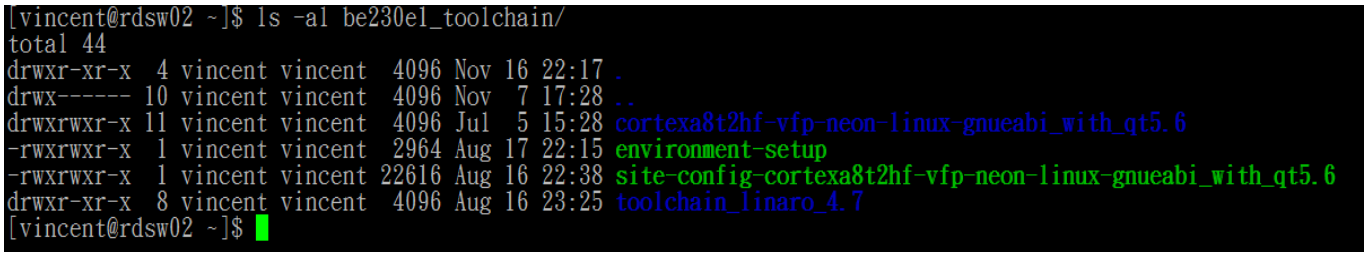
Download SDK/Toolchain

Get the SDK and tool chain from Bolymin download link. Use this tool chain to build executable programs for BE230E1 Linux platform.

After downloading the SDK/toolchain file to your Linux host, unzip and untar this tar.bz2 file by

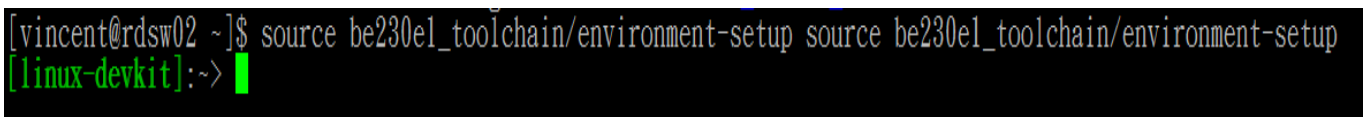
```
$tar xvfj be230e1_toolchain.tar.bz2
```

to be located at some path you assigned.



Use SDK/Toolchain

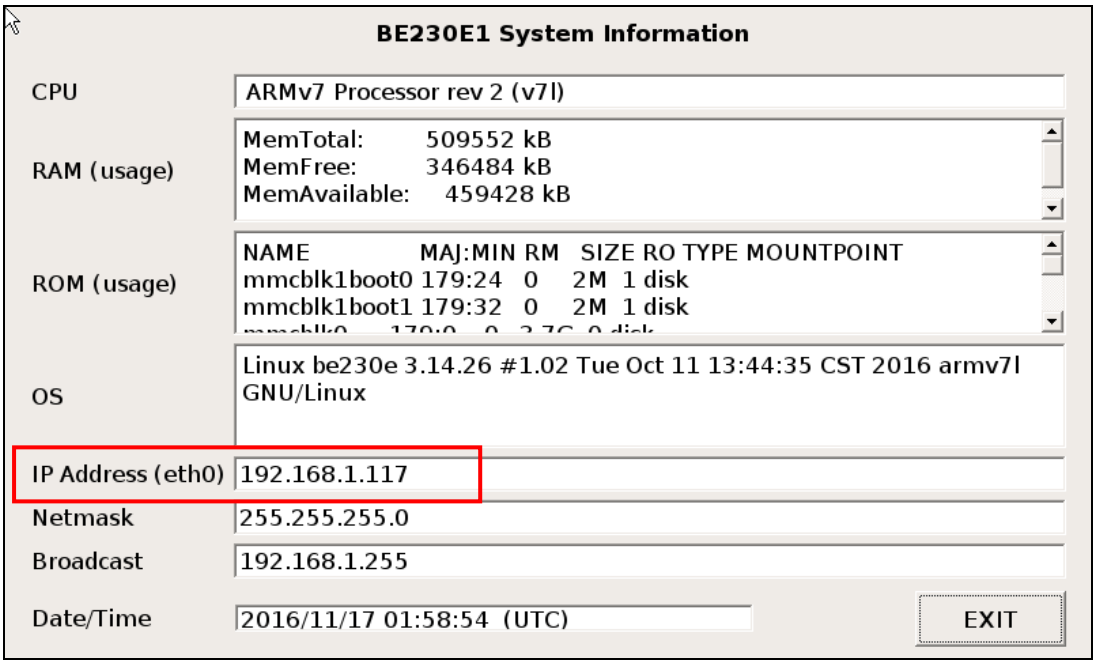
Before starting to build your applications, please execute **environment-setup** script file to setup/export several environment variables for cross compiler/linker or Qt libraries.



1.3 Connect BE230E1 Device to PC

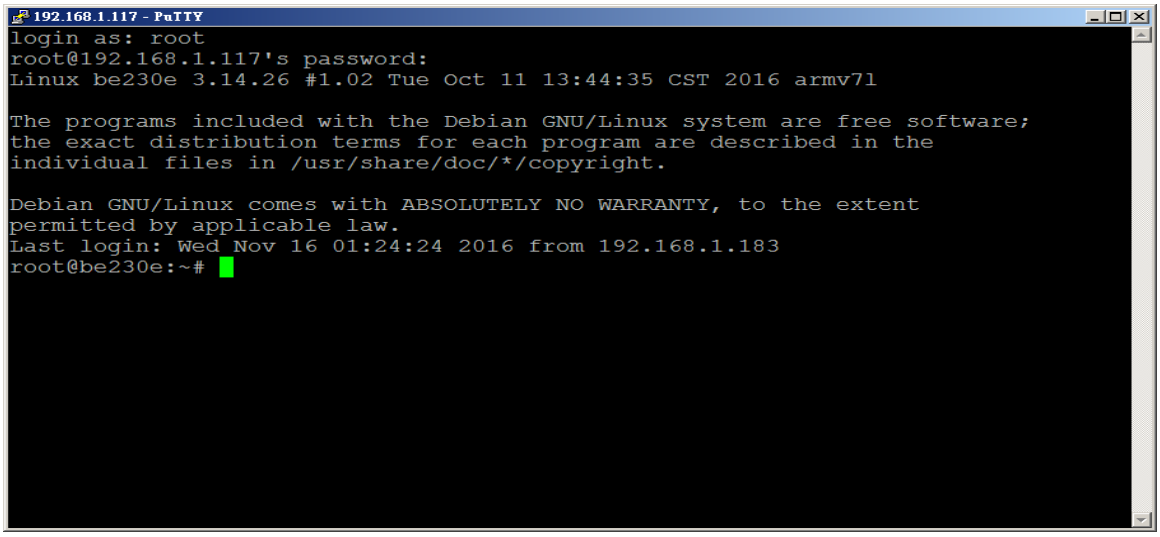
0. Get IP address of LAN port

Assume there's DHCP server in your network environment, you can get current IP address of LAN port after plugging RJ45 cable into BE230E1 and then run 'System Info' test app.



1. Telnet connection (thru LAN or Wi-Fi)

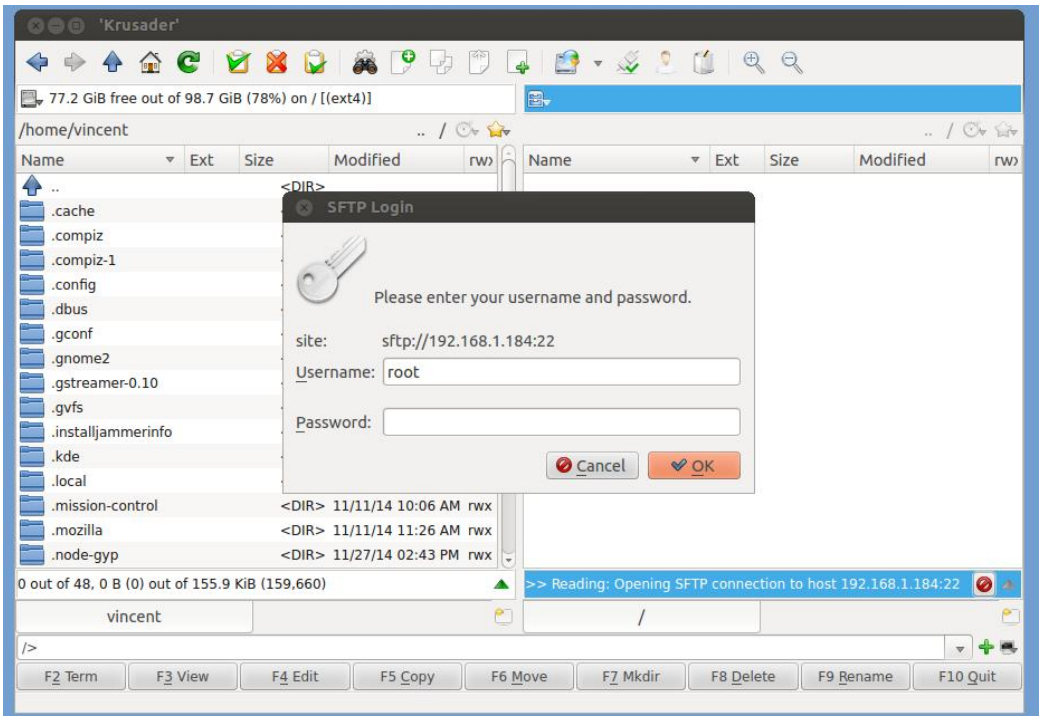
The BE230E1 device has a LAN and WLAN interface so that you can do remote connection to login the device thru network by some telnet client software like SSH.



Default login account is "**root**", password is "**root**".

2. Put your app or files to the BE230E1 device

To transfer your built Apps or other files to the BE230E1 device, the easy and fast way is to use the file transfer software such as SFTP or SCP. Thus you could transfer files between PC (Windows or Linux) and BE230E1 target device.



Login account is “root”, password is “root”.

2 Test Apps for BE230E1

On this BE230E1 device, there are several test apps written in Qt/C++ and/or QML/QtQuick that can be used to simply verify the functionalities of corresponding hardware module or interfaces. In addition, we've offered Qt 4.6 framework porting on the board and it could be a choice for you to write your own GUI application on BE230E1 device.

Test App Items	Description
GPIO	To verify 8 GPIO ports functionality. Written in Qt/C++ code.
ADC	To verify 4 ADC channels functionality. Written in Qt/C++ code.
UART	To verify UART functionality, used for RS232/RS485 connection. Written in Qt/C++ code.
Bluetooth	To verify Bluetooth basic functions, like scanning and pairing/unpairing. Written in Qt/C++ code.
WLAN	To verify WLAN functionality. Written in Qt/C++ code.
LAN	To verify accessing network thru LAN port. Written in Qt/C++ code.
LCD	To verify LCD display. Written in Qt/C++ and QML/QtQuick code.
Backlight	To verify screen backlight brightness. Written in Qt/C++ and QML/QtQuick
Touch Panel	To verify touch panel functionality. Written in Qt/C++ code.
Speaker	To verify speaker functionality. Written in Qt/C++ code.

2.1 GPIO Test

This test app is used to test built-in 8 GPIO pins on BE230E1 device. Users can change both the direction (either input or output) and value (either high or low) of these 8 pins by API functions provided by Bolymin. Developers could refer to our sample code for more details.

2.1.1 GPIO Pins Definition

The value of 8 GPIO pins is located in below list. It could only set as "**0**" (meaning LOW) or "**1**" (meaning HIGH). The default for all pins is "**1**" after reboot.

```
static QString const SYSFS_GPIO_PINS_VALUE[] =
                     {
                      "/sys/class/gpio/gpio48/value",   // Pin 1
                      "/sys/class/gpio/gpio7/value",    // Pin 2
                      "/sys/class/gpio/gpio53/value",   // Pin 3
                      "/sys/class/gpio/gpio54/value",   // Pin 4
                      "/sys/class/gpio/gpio67/value",   // Pin 5
                      "/sys/class/gpio/gpio61/value",   // Pin 6
                      "/sys/class/gpio/gpio68/value",   // Pin 7
                      "/sys/class/gpio/gpio66/value"    // Pin 8
                     };
```

The direction of 8 GPIO pins is located in below list. It could only set as "**in**" or "**out**".
The default for all pins is "**in**" after reboot.

```
static QString const SYSFS_GPIO_PINS_DIRECTION[] =
                     {
                      "/sys/class/gpio/gpio48/direction",   // Pin 1
                      "/sys/class/gpio/gpio7/direction",    // Pin 2
                      "/sys/class/gpio/gpio53/direction",   // Pin 3
                      "/sys/class/gpio/gpio54/direction",   // Pin 4
                      "/sys/class/gpio/gpio67/direction",   // Pin 5
                      "/sys/class/gpio/gpio61/direction",   // Pin 6
                      "/sys/class/gpio/gpio68/direction",   // Pin 7
                      "/sys/class/gpio/gpio66/direction"    // Pin 8
                     };
```

2.1.2 Bolymin GPIO API

Source: gpioaccess.c
Header: gpioaccess.h

Function Syntax	int GPIO_Get_Pin_Direction(int pinNum)	
Description	To get the GPIO pin direction by assigned pin number.	
Parameters	pinNum	Pin number, between 1 and 8.
Return Value	1	It means input direction
	2	It means output direction
	0	Undefined

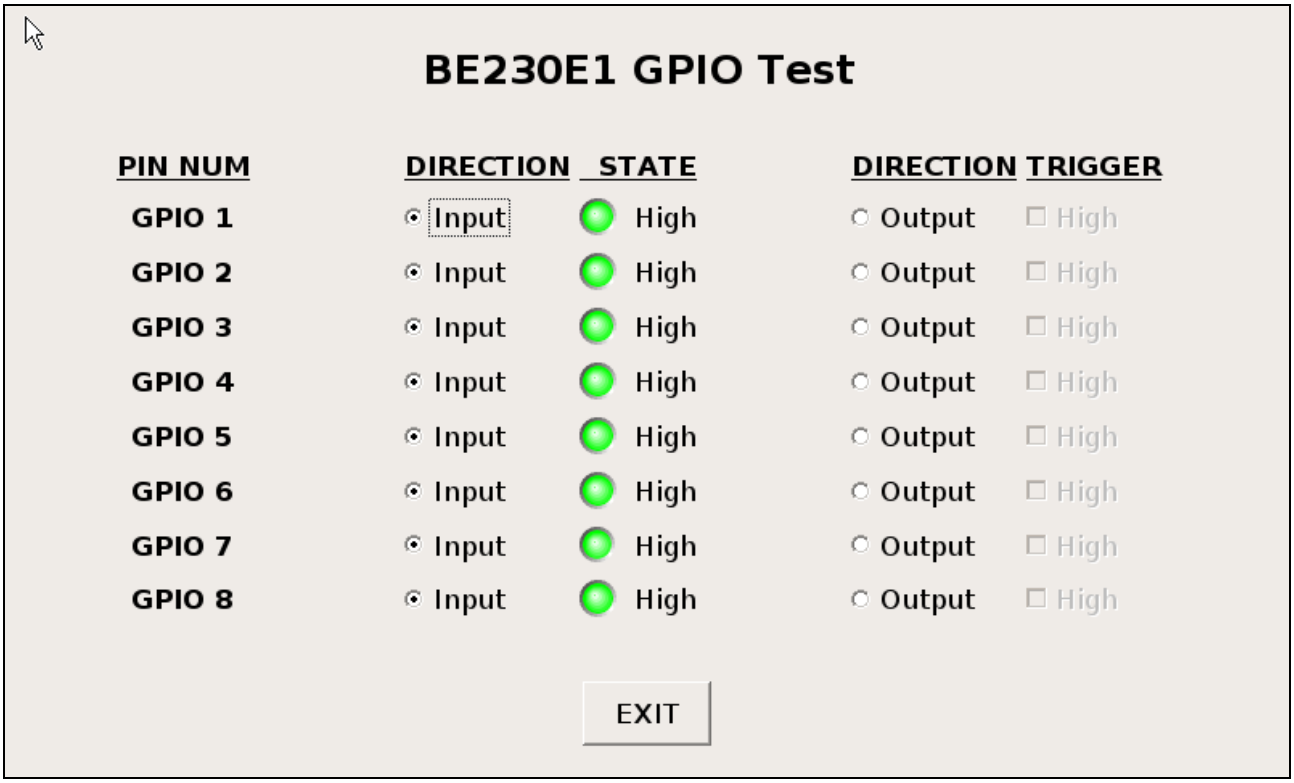
Function Syntax	void GPIO_Set_Pin_Direction(int pinNum, bool bInput)	
Description	To set the GPIO pin direction by assigned pin number.	
Parameters	pinNum	Pin number, between 1 and 8.
	bInput	'true' is for input direction; 'false' is for output direction
Return Value	none	

Function Syntax	int GPIO_Get_Pin_Value(int pinNum)	
Description	To get the GPIO pin value by assigned pin number.	
Parameters	pinNum	Pin number, between 1 and 8.
Return Value	0	It means 'Low' state
	1	It means 'High' state
	others	Undefined state

Function Syntax	void GPIO_Set_Pin_Value(int pinNum, bool bPullLow)	
Description	To set the GPIO pin value by assigned pin number and boolean type value to set low or high.	
Parameters	pinNum	Pin number, between 1 and 8.
	bPullLow	'true' means Low state 'false' means High state
Return Value	none	

2.1.3 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.



2.2 ADC Test

This test app is used to test the four ADC channels of BE230E1. The resolution of all ADC channels is **12-bit** and each value range is from 0 volt to 3.3 volt.

2.2.1 ADC Channels Definition

```
static QString const SYSFS_ADC_CHANNELS[] =
{
  "/sys/bus/iio/devices/iio\:device0/in_voltage4_raw",    // ADC CH1
  "/sys/bus/iio/devices/iio\:device0/in_voltage5_raw",    // ADC CH2
  "/sys/bus/iio/devices/iio\:device0/in_voltage6_raw",    // ADC CH3
  "/sys/bus/iio/devices/iio\:device0/in_voltage7_raw",    // ADC CH4
};
```

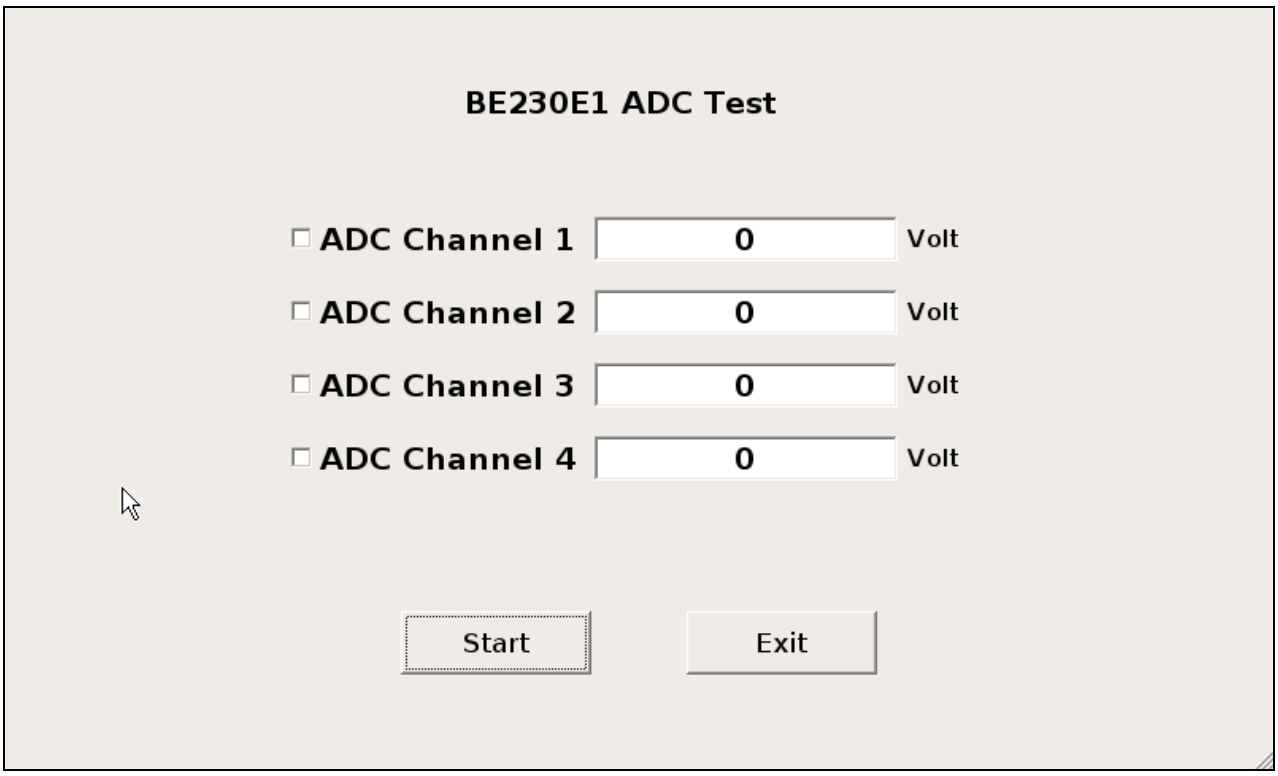
2.2.2 Bolymin ADC API

Source: adcreader.c
Header: adcreader.h

Function Syntax	float Read_Adc_Value(unsigned int chNum)	
Description	To read current ADC value by assigned channel number.	
Parameters	chNum	Channel number, from 1 to 4
Return Value	ADC value in voltage, ranged from 0 ~ 3.3 volt.	

2.2.3 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.



2.3 UART Test

This test app is modified from Qt's Terminal sample of Serial Port Example, which shows the main features of the QSerialPort class, like configuration, I/O implementation and so forth.

2.3.1 UART Port Definition

On BE230E1, UART ports can be used for RS232 or RS485. Please see the user manual for the detailed pin definition of Multi I/O cable.

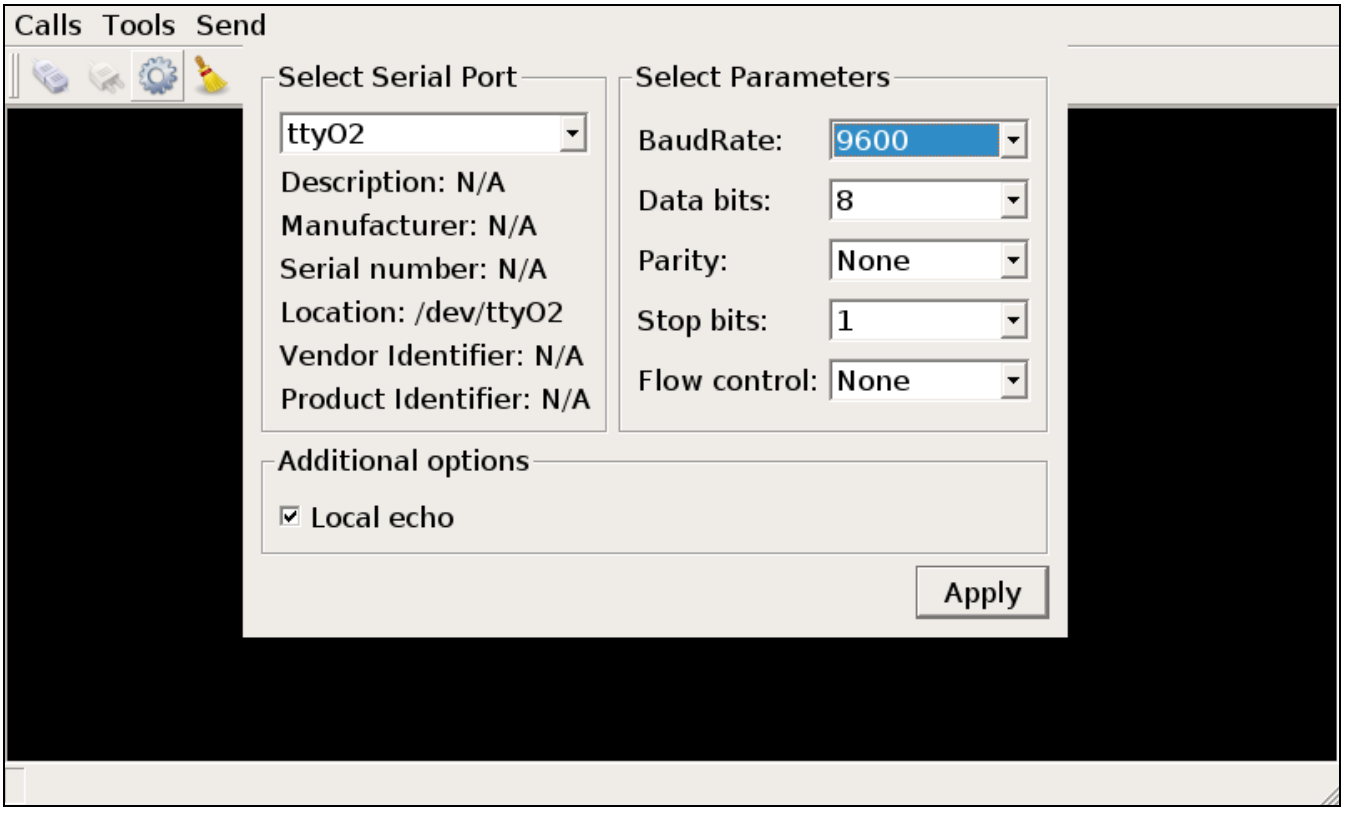
Device Name	Description
/dev/ttyO2	For RS232
/dev/ttyO4	For RS422/RS485 (selected by cable)

2.3.2 Qt Serial Port API

Please refer to Qt Serial Port C++ Classes for more details.

2.3.2 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.



2.4 Bluetooth Test

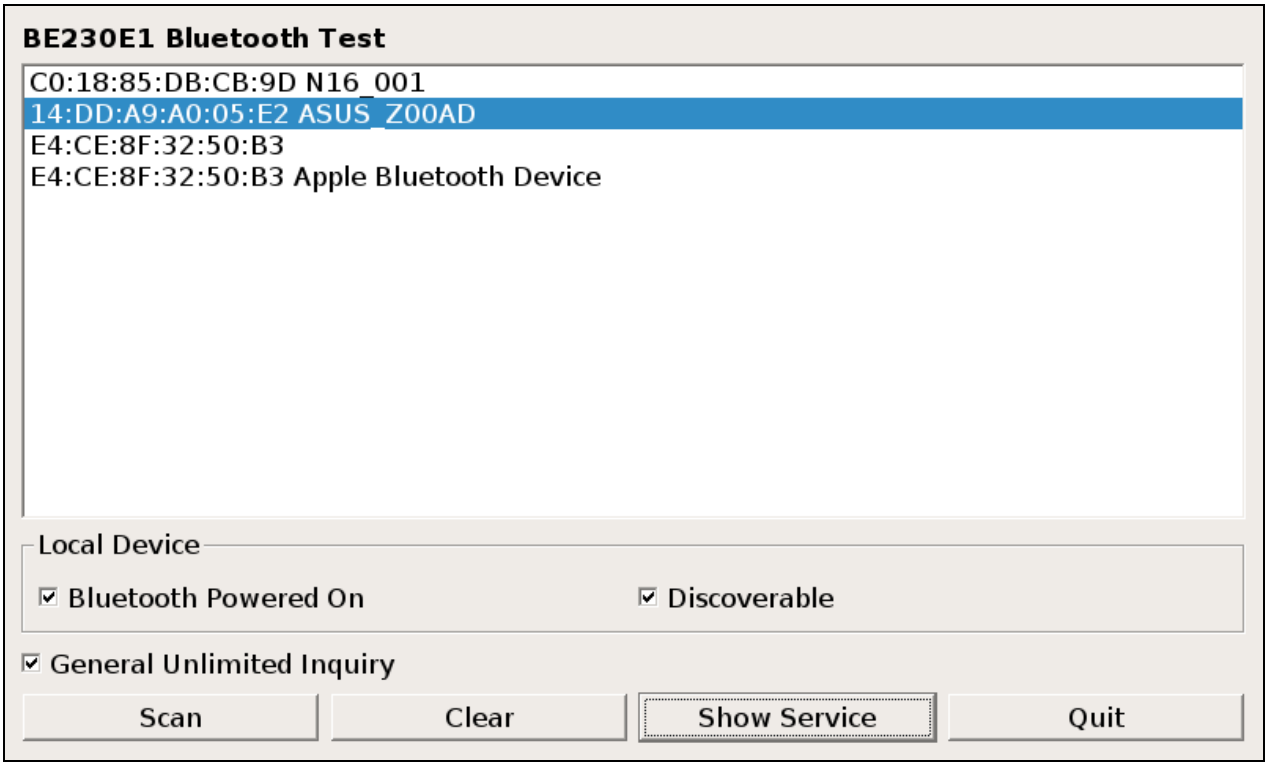
This test app is modified from Qt's Bluetooth Scanner example. It could be used to scan the available Bluetooth devices nearby and do pairing/unpairing to assigned remote device.

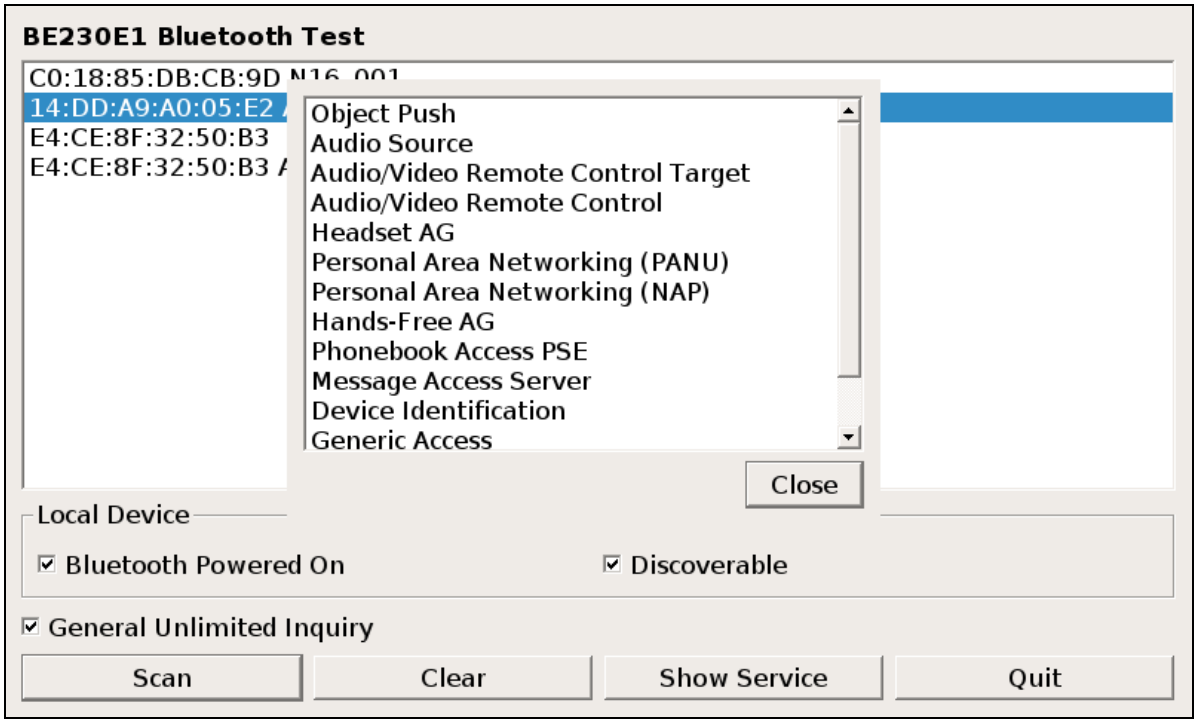
2.4.1 Qt Bluetooth API

This example utilizes several Qt Class of Bluetooth related like QBluetoothDeviceInfo, QBluetoothLocalDevice, and QBluetoothDeviceDiscoveryAgent.

2.4.2 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.




BE230E1 Bluetooth Test
C0:18:85:DB:CB:9D N16_001
14:DD:A9:A0:05:E2
E4:CE:8F:32:50:B3
E4:CE:8F:32:50:B3
Object Push
Audio Source
Audio/Video Remote Control Target
Audio/Video Remote Control
Headset AG
Personal Area Networking (PANU)
Personal Area Networking (NAP)
Hands-Free AG
Phonebook Access PSE
Message Access Server
Device Identification
Generic Access
Close
Local Device
Bluetooth Powered On
Discoverable
General Unlimited Inquiry
Scan
Clear
Show Service
Quit

2.5 WLAN Test

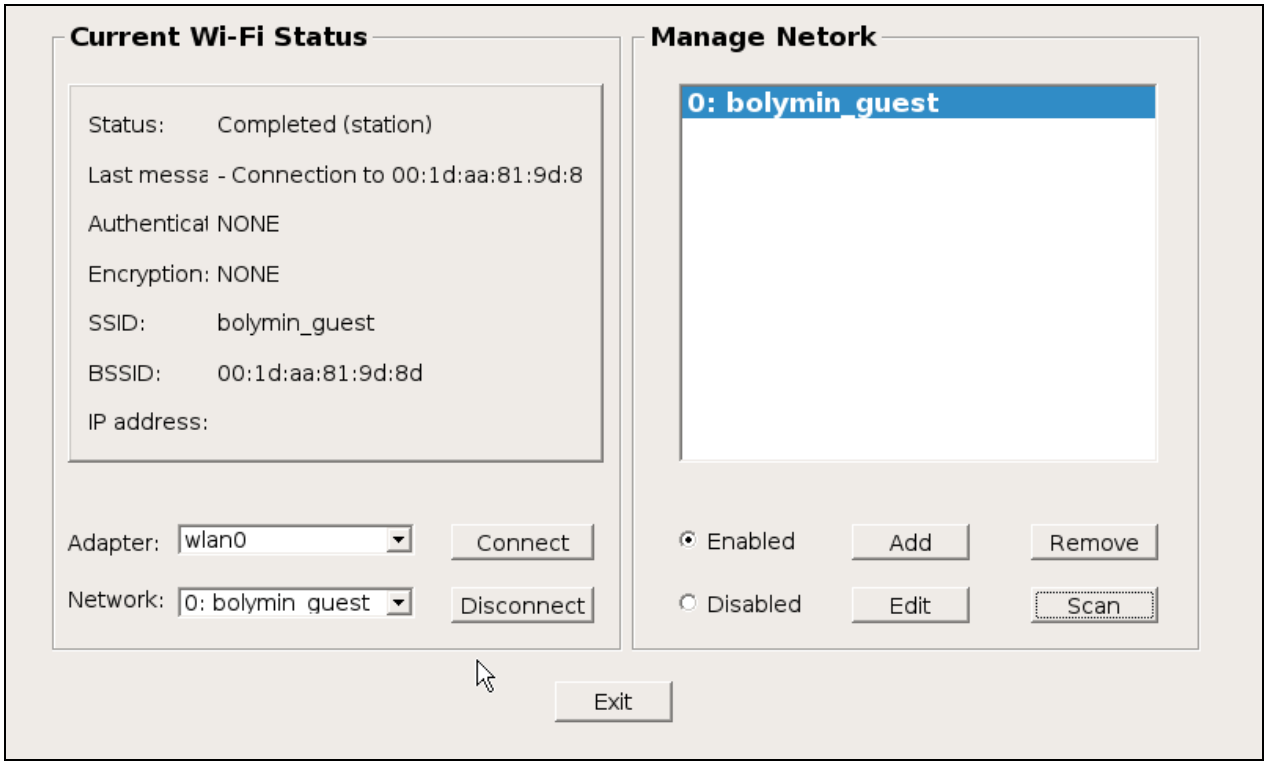
This example can be used to verify the wireless LAN functionality of BE230E1.

2.5.1 Sample Code Source

This test app is the front-end GUI part (wpa_gui-qt4) from wpa_supplicant and we've modified it only for simple testing.

2.5.2 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.



2.6 LAN Test

2.6.1 Description

This test app uses a QWebView class acting as a web browser to access network.

2.6.2 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.

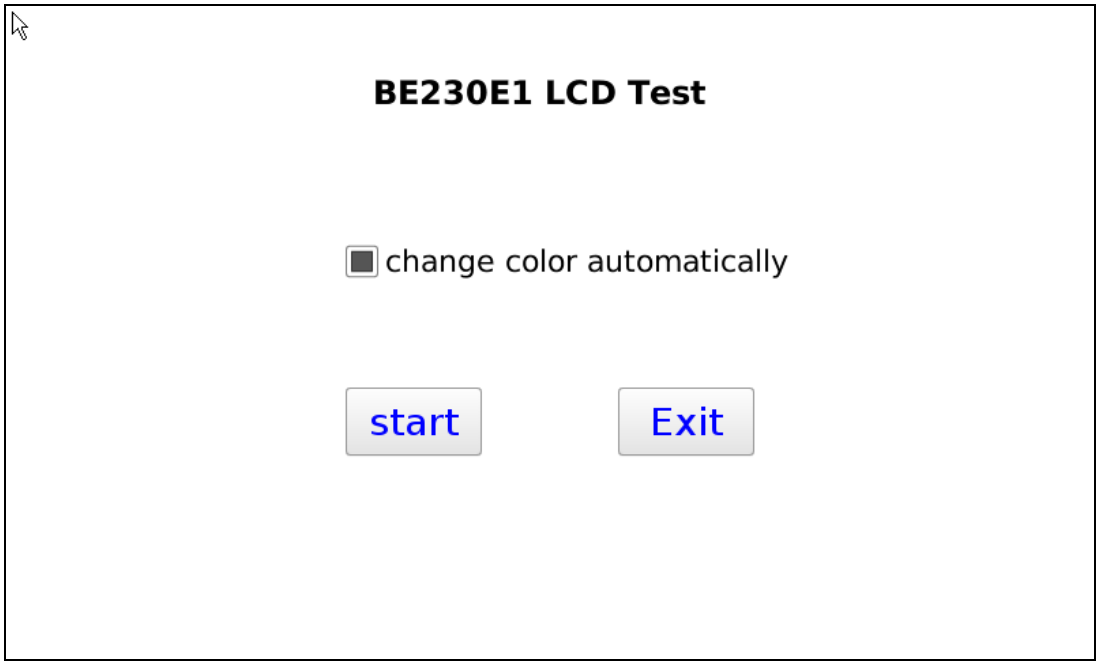
2.7 LCD Test

This app is used to test the LCD screen by showing different colors (Red, Green, Blue, Black, and White) sequentially.

2.7.1 QML

This app is written in QML primarily. Please find main.qml in LcdTest source example for more details.

2.7.2 GUI



2.8 Backlight Control

This app is used to control the screen backlight brightness ranged from 1 to 8.

2.8.1 Backlight Brightness Value

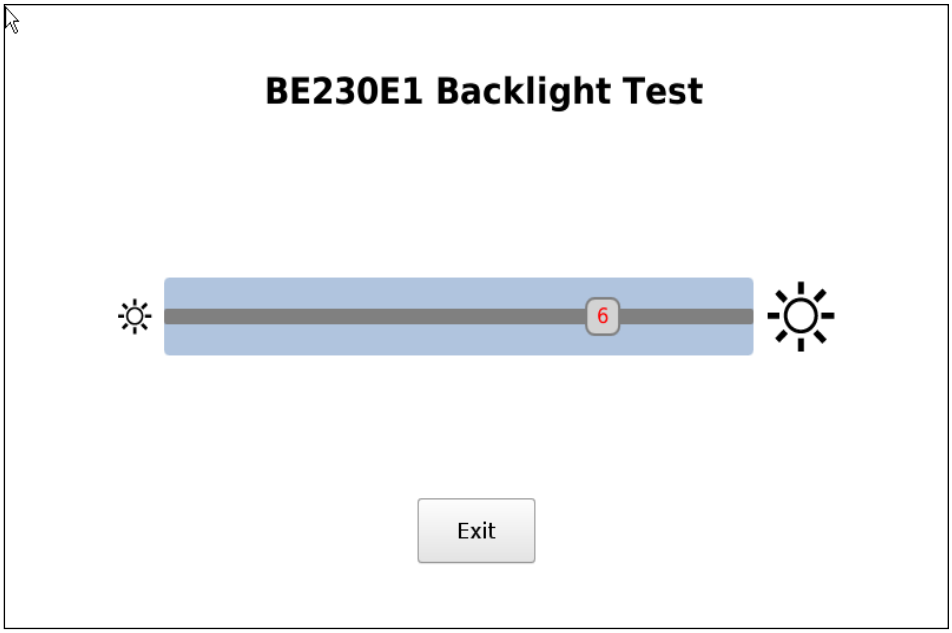
The exposed Linux kernel parameter to control brightness value is located at "/sys/class/backlight/backlight.8/brightness".
The available configured value is between 0 and 8. Note that the screen would be full black if you set brightness to be 0.

2.8.2 Bolymin Backlight API

Function Syntax	int getBrightness()
Description	Get backlight brightness value
Parameters	none
Return Value	brightness value : between 1 and 8

Function Syntax	void setBrightness(int value)	
Description	Set backlight brightness value	
Parameters	value	Between 1 and 8
Return Value	none	

2.8.3 GUI



2.9 Touch Panel Test

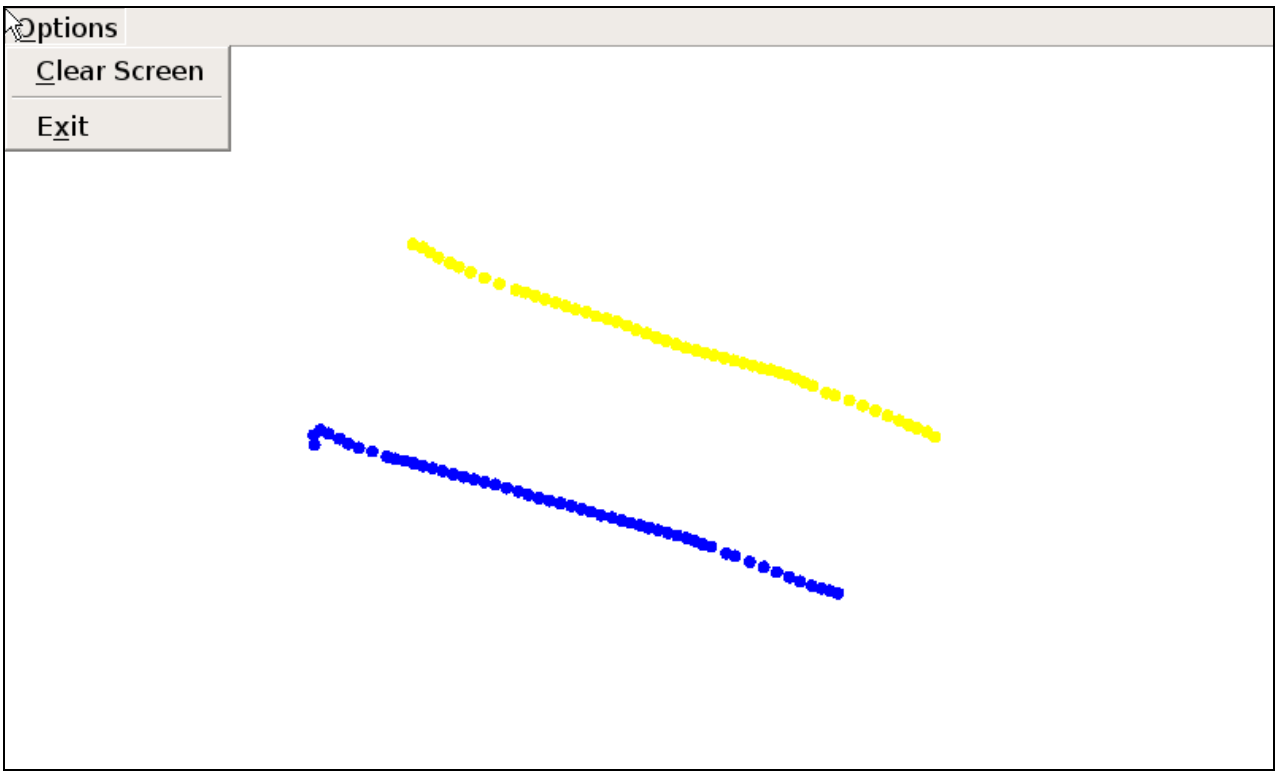
This test app is modified from Qt's Scribble Example and could be used to draw lines while user's figure is pressing and moving on the screen.

2.9.1 Qt C++ Class

This example utilizes QTouchEvent related class of Qt to detect user's touch motion on the screen. Programmer could visit Qt's website to see more detailed information.

2.9.2 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.



2.10 Speaker Test

This app is modified from Qt's Media Player example that can be used to play the audio file (e.g. MP3) to test the built-in speaker of BE230E1.

2.10.1 Qt C++ Class

The example uses a QMediaPlayer object to control the play/pause/stop functions. To give the application playlist capability it also uses a QMediaPlayList object.

2.10.2 GUI

Developers could load the sample code by Qt Creator to open the GUI layout file.